## REMARKS

This is a full and timely response to the non-final Office Action (Paper No. 2) mailed by the U.S. Patent and Trademark Office on September 27, 2002. Upon entry of the attached amendments, claims 1-12 have been amended, claims 13, 14, and 15 have been canceled without prejudice, waiver, or disclaimer. Claims 1-12 are hereby amended to more particularly point out and distinctly claim the subject matter that Applicant regards as the invention. No new matter is added. Reconsideration of the pending claims is respectfully requested, in view of the changes as denoted in the attached appendix and the following remarks. Each objection and rejection presented in the Office Action is discussed in the remarks that follow.

Note: Appendix A - Annotated Claims
Appendix B - Substitute Specification and Abstract
Appendix C - Annotated Specification and Abstract
Appendix D - Redlined Drawings
Appendix E - Substitute Drawings

### I. Objection to the Drawings

#### A. Statement of the Objection

The drawings are objected to for allegedly failing to comply with 37 C.F.R. 1.84(p)(5) because they do not include reference numerals mentioned in the description and because they include other reference numerals not mentioned in the description.

#### B. Discussion of the Objection - Drawings

Proposed drawing corrections are included in Appendix D for review and acceptance by the Examiner. Applicant has amended the Drawings to include reference numerals for items discussed in the detailed description and amended the Drawings to remove reference numerals from items not discussed in the specification. Accordingly, upon entry of the amendments and acceptance of the proposed substitute drawings as included in Appendix E, the objection is rendered moot.

**II. Objection to the Specification - Title**

**A. Statement of the Objection**

The title of the disclosure is objected to for allegedly being non-descriptive of the subject matter of the claims.

**B. Discussion of the Objection - Title**

Applicant has amended the title in the attached amendments. Consequently, upon entry of the attached amendments, the objection to the title is rendered moot.

**III. Objection to the Specification - Abstract**

**A. Statement of the Objection**

The abstract of the disclosure is objected to for allegedly containing acronyms without spelling out the words associated with the acronym.

**B. Discussion of the Objection - Abstract**

Applicant has amended the abstract in the attached amendments. Accordingly, upon entry of the attached amendments, the objection to the abstract is rendered moot.

**IV. Claim Objections - Claims 3 and 5**

**A. Statement of the Objection**

Pending claims 3 and 5 are objected to for allegedly failing to comply with 35 U.S.C. §112, Second Paragraph. Specifically, claim 3 recites the limitation "substantially currently." Claim 5 recites the limitation "release."

**B. Discussion of the Objection – Claims 3 and 5**

Applicant has amended claims 3 and 5 to more particularly point out and distinctly claim the subject matter that Applicant regards as the invention. Specifically, Applicant has amended claim 3 to recite the limitation "substantially concurrently." Furthermore, Applicant has amended claim 5 such that the limitation "release" is grammatically correct in the claim. Consequently, upon entry of the attached claim amendments, the objection to claims 3 and 5 is rendered moot.

**V. Claim Objections - Claims 9 and 10**

**A. Statement of the Objection**

Pending claims 9 and 10 are objected to for allegedly containing two (2) semicolons after the limitation "the step of."

**B. Discussion of the Objection – Claims 3 and 5**

Applicant has amended claims 9 and 10 to correct these typographical errors. Consequently, upon entry of the attached claim amendments, the objection to claims 9 and 10 is rendered moot.

**VI. Response to 35 U.S.C. §102 Rejections – Claims 1-3**

**A. Statement of the Rejection**

Claims 1-3 presently stand rejected under 35 U.S.C. §102(e) as allegedly being anticipated by Roussel et al. (U.S. Patent Number 6,230,257 "the '257 patent.")

**B. The '257 Patent**

The '257 patent is directed to a method and apparatus for staggering execution of an instruction. According to one embodiment of the invention, a single macro instruction is received wherein the single macro instruction specifies at least two logical registers and wherein the two logical registers respectively store a first and second packed data operands having corresponding data elements. An operation specified by the single macro instruction is then performed independently on a first and second plurality of the corresponding data elements from said first and second packed data operands at different times using the same circuit to independently generate a first and second plurality of resulting data elements. The first and second plurality of resulting data elements are stored in a single logical register as a third packed data operand. (U.S. Patent 6,230,257 column 2, lines 26-40).

**C. Discussion of the Rejection**

Applicant respectfully traverses the rejection of claims 1-3 under 35 U.S.C. §102(e) for at least the reason that the cited reference fails to disclose, teach, or suggest each element in the claims. It is axiomatic that "[a]nticipation requires the disclosure in a single prior art reference of *each element* of the claim under consideration." *W.L. Gore & Associates, Inc. v. Garlock, Inc.*, 721 F.2d 1540, 1554,

220 U.S.P.Q. 303, 313 (Fed. Cir. 1983) (*emphasis added*). Therefore, every claimed feature of the claimed invention must be represented in the applied reference (the '257 patent) to constitute a proper rejection under 35 U.S.C. § 102(e).

For convenience of analysis, independent claim 1, as amended, is repeated below in its entirety.

> 1. An apparatus for performing single-instruction multiple-data instructions using a single multiply-accumulate (MAC) unit, comprising:
> ***a MAC unit configured to generate a data result***, the data result having a first half and a second half;
> ***a register communicatively coupled to the MAC unit, the register configured to store the first half of the data result***; and
> ***a miscellaneous-logic unit configured to initiate the release of the first half of the data result from the register to synchronize the first half of the data result with the second half of the data result.***

(Applicant's independent claim 1 - *emphasis added.*)

Applicant respectfully asserts that the cited art of record fails to disclose, teach, or suggest at least the emphasized elements of pending claim 1 as shown above. Consequently, claim 1 is allowable.

In this regard, the Office alleges, "Roussel et al. have taught an apparatus for performing single instruction multiple data instructions using a single multiply accumulate (MAC) unit . . .said MAC unit generates a first half of a data result and a second half of a data result (Figure 3, column 3, lines 21-43, the first half, or low order, of the data result is (x1+y1)(x0+y0), the second half, or higher order, of a data result is (x3+y3)(x2+y2), which are both generated from a MAC unit); . . ." Applicant disagrees that the '257 patent teaches using a single MAC unit to perform a SIMD operation. The Office Action's interpretation of what is allegedly disclosed in the '257 patent is in direct contradiction with the following statement from the '257 patent.

> Thus, according to an embodiment of the present invention, ***execution units*** *are provided that contain only half the hardware* ***(e.g. two single precision ADD execution units and two single precision MUL execution units)***, instead of the execution units required to process the full width of the operands in parallel as found in a current processor.

(U.S. Patent 6,230, 257, column 4, lines 59-64, *emphasis added.*)

The system apparently disclosed in the '257 patent describes two single-precision ADD-execution units and two single-precision MUL-execution units. Two single-precision ADD-execution units and two single-precision MUL-execution units *are not* a single MAC unit as the Office Action rejection suggests. While column 3, lines 39-41 indicates that other execution units such as FMAC units may be used, the '257 patent fails to describe how a single MAC unit can be used to perform a SIMD operation. Consequently, for at least this reason, the '257 patent fails to disclose, teach, or suggest each element of the Applicant's claimed invention.

Furthermore, Figures 4A and 5 of the '257 patent illustrate the use of delay elements (M1, M2, and M3 in Figure 4A and M 190, M 200, and M 400 in Figure 5) apparently for delaying the transmission of a portion of a data word to an execution unit. Clearly, the apparatus apparently disclosed in the '257 patent is different from Applicant's claimed invention in this regard. The apparatus of the '257 patent uses passive delay elements arranged to intercept data as it is clocked to the plurality of execution units. In contrast, Applicant's claimed invention recites "***a MAC unit***" and a "***a register communicatively coupled to the MAC unit, the register configured to store the first half of the data result; . . .***" Passive delay units arranged to intercept data bytes on their way to an execution unit *do not* teach a register coupled to a MAC unit configured to store a data result. Consequently, for at least this second reason, the '257 patent fails to disclose, teach, or suggest each element of the Applicant's claimed invention.

Moreover, the apparatus of the '257 patent *does not* disclose the claimed "***miscellaneous-logic unit configured to initiate the release of the first half of the data result from the register to synchronize the first half of the data result with the second half of the data result.***" As noted above, the apparatus of the '257 patent apparently discloses the use of passive delay elements to delay data bytes before they

are processed by multiple execution units. In further contrast, Applicant's claimed invention uses "a ***miscellaneous-logic unit configured to initiate the release of the first half of the data result from the register to synchronize the first half of the data result with the second half of the data result.***" Thus, the '257 patent fails to disclose, teach, or suggest each element of the Applicant's claimed invention.

Consequently, for at least these reasons, Applicant respectfully submits that Applicant's claim 1 and each dependent claim that depends therefrom are allowable over the '257 patent. Applicant therefore requests that the rejection of claims 1-3 be withdrawn.

## VII. Response to 35 U.S.C. §103 Rejections – Claims 4-15

### A. Statement of the Rejection

Claims 4-15 presently stand rejected under 35 U.S.C. §103(a) as allegedly being unpatentable over Roussel et al. (U.S. Patent No. 6,230,257) "the '257 patent") in view of Phillips et al. (U.S. Patent No. 6,038,652, "the '652 patent.") Concerning claim 4 the Office Action rejection acknowledges that the '257 patent is silent regarding a miscellaneous logic unit generating an exception result when the miscellaneous logic unit determines that the first half of the data result is in error. To remedy this admitted failure of the '257 patent to disclose, teach, or suggest each element of the claimed invention, the Office Action alleges the '652 patent teaches that overflow or other exceptions may occur during multiply and accumulate functions and that it is necessary to report the exception so that appropriate action may be taken. The rejection then concludes it would have been obvious to one skilled in the art at the time of the invention to modify the system of the '257 patent to generate and report an exception.

### B. The '652 Patent

The '652 patent is directed to a method and apparatus for reporting exception in a single instruction multiple data (SIMD) processor in computing an arithmetic function for a plurality of argument data. The SIMD processor is configured for processing N elements simultaneously. A sequence of instructions is re-arranged to allocate the plurality of argument data in the N elements. The N elements are processed simultaneously. The exceptions for N elements are detected

simultaneously. The detected exceptions are then combined to generate a global exception. (U.S. Patent 6,038,652, Abstract.)

**C. Discussion of the Rejection**

**1. Claims 4 and 5**

Applicant respectfully traverses the rejection of claims 4 and 5 under 35 U.S.C. §103(a) for at least the reason that even the proposed combination of references fails to disclose, teach, or suggest each element in the claims.

As pointed out above, the '257 patent fails to teach element of Applicant's independent claim 1. Thus, modifying the apparatus of the '257 patent to add exception handling as the Office Action as suggests fails to remedy the failure of the '257 patent to disclose, teach, or suggest Applicant's claimed apparatus. Because independent claim 1 is allowable, dependent claims 4 and 5 are also allowable. *See In re Fine*, 837, F.2d 1071, 5 U.S.P.Q.2d 1596, 1598 (Fed. Cir. 1988). Accordingly, Applicant respectfully requests that the rejection of claims 4 and 5 be withdrawn.

**2. Claims 6-10**

Applicant respectfully traverses the rejection of claims 6-10 under 35 U.S.C. §103(a) for at least the reason that even the proposed combination of references fails to disclose, teach, or suggest each method step in the claims.

For convenience of analysis, independent claim 6, as amended, is repeated below in its entirety.

> 6. A method for performing single-instruction multiple-data instructions using a single multiply-accumulate unit, comprising:
> providing a multiply-accumulate unit configured to generate a first half of a data result and a second half of a data result;
> applying the first half of the data result at an input of a register;
> using a miscellaneous-logic unit to generate an exception result; and
> ***applying the first half of the data result and the second half of the data result at an input of a buffer when the miscellaneous-logic unit determines that the first half of the data result and the second half of the data result are valid, otherwise applying an exception***

> ***result at the input of the buffer when the miscellaneous-logic unit determines that said] the first half of the data result and the second half of the data result are invalid.***

(Applicant's independent claim 6 - *emphasis added.*)

Applicant respectfully asserts that the cited art of record fails to disclose, teach, or suggest at least the emphasized method steps of pending claim 6 as shown above. Consequently, claim 6 is allowable.

As acknowledged by the Office Action rejection of claim 4, the '257 patent is silent regarding detecting and reporting exception(s). To remedy this failure, the Office Action rejection alleges that one skilled in the art could reach the Applicant's claimed invention by combining the exception detection and handling as taught in the '652 patent. Applicant respectfully disagrees. The apparatus apparently disclosed in the '652 patent shows a plurality of exception detectors in a one-to-one relationship with a plurality of processing elements. When exceptions are indicated by the exception detectors, an exception signal is forwarded to a combining unit that generates a global exception from the one or more exception signals. Significantly, the '652 patent *does not* describe what to do with the global exception after it is generated. In contrast, Applicant's claimed invention recites "***applying the first half of the data result and the second half of the data result at an input of a buffer when the miscellaneous-logic unit determines that the first half of the data result and the second half of the data result are valid, otherwise applying an exception result at the input of the buffer when the miscellaneous-logic unit determines that said] the first half of the data result and the second half of the data result are invalid.***" Neither the '257 patent nor the '652 patent disclose, teach, or suggest the emphasized method step. Consequently, even the proposed combination of the '257 patent and the '652 patent fail to disclose, teach, or suggest each method step recited in Applicant's claimed invention.

Accordingly, for at least these reasons, Applicant's claim 6 and each dependent claim that depends therefrom are allowable over the proposed combination of the '257 patent and the '652 patent. Applicant therefore requests that the rejection of claims 6-10 be withdrawn.

**3. Claims 11 and 12**

Applicant respectfully traverses the rejection of claims 11 and 12 under 35 U.S.C. §103(a) for at least the reason that even the proposed combination of references fails to disclose, teach, or suggest each element in the claims.

For convenience of analysis, independent claim 11, as amended, is repeated below in its entirety.

> 11. An apparatus for performing single-instruction multiple-data instructions, comprising:
> means for generating a first data result responsive to a first operand;
> means for storing the first data result;
> means for generating a second data result responsive to a second operand;
> means for generating an exception result responsive to the first and second data results;
> ***means for forwarding the first data result and the second data result to a buffer when the exception result indicates that the first data result and the second data result are valid***; and
> ***means for communicating the exception to the buffer when the means for forwarding indicates that the first and second data results are invalid.***

(Applicant's independent claim 11 - *emphasis added.*)

Applicant respectfully asserts that the cited art of record fails to disclose, teach, or suggest at least the emphasized elements of pending claim 11 as shown above. Consequently, claim 11 is allowable.

In this regard, the Office Action rejection alleges that claim 11 does not recite limitations above the claimed invention set forth in claim 6. Applicant disagrees with the rejection for at least the reason that the proposed combination of the '257 patent and the '652 patent fail to disclose, teach, or suggest each element recited in Applicant's claimed invention. As noted above, the '652 patent *does not* describe what to do with the global exception after it is generated. In contrast, Applicant's claimed invention recites "***means for forwarding the first data result and the second data result to a buffer when the exception result indicates that the first data result and the second data result are valid***; and ***means for communicating the exception to the buffer when the means for forwarding indicates that the first and second data results are invalid.***" Neither the '257 patent nor the '652 patent disclose, teach, or

suggest "***means for forwarding the first data result and the second data result to a buffer when the exception result indicates that the first data result and the second data result are valid***; and ***means for communicating the exception to the buffer when the means for forwarding indicates that the first and second data results are invalid.***" Consequently, even the proposed combination of the '257 patent and the '652 patent fail to disclose, teach, or suggest each element recited in Applicant's claimed invention.

Accordingly, for at least these reasons, Applicant's claim 11 and claim 12 that depends therefrom are allowable over the proposed combination of the '257 patent and the '652 patent. Applicant therefore requests that the rejection of claims 11 and 12 be withdrawn.

## CONCLUSION

In summary, Applicant respectfully requests that all outstanding claim rejections be withdrawn. Applicant respectfully submits that presently pending claims 1-12 are allowable and the present application is in condition for allowance. Accordingly, a Notice of Allowance is respectfully solicited. Should the Examiner have any comments regarding the Applicant's response or intends to dispose of this matter in a manner other than a Notice of Allowance, Applicant requests that the Examiner telephone Applicant's undersigned attorney.

Respectfully submitted,

**THOMAS, KAYDEN, HORSTEMEYER & RISLEY, L.L.P.**

By: Robert A. Blaha

Robert A. Blaha
Registration No. 43,502

**THOMAS, KAYDEN, HORSTEMEYER & RISLEY, L.L.P.**
100 Galleria Parkway, Suite 1750
Atlanta, Georgia 30339-5948
(770) 933-9500

## APPENDIX C - AMENDMENTS TO THE SPECIFICATION

The following specification has been amended by deleting the bracketed ("[ ]") portions and adding the underlined ("__") portions.

# AN APPARATUS AND METHOD FOR [COMBINING SIMD RESULTS USING A STALL BASE METHODOLOGY] PERFORMING SINGLE-INSTRUCTION MULTIPLE-DATA INSTRUCTIONS

## TECHNICAL FIELD

[0001] The present invention is generally related to performing single-instruction multiple data (SIMD) [SIMD (Single Instruction Multiple Data)] instructions and, more particularly, is related to an apparatus and method for performing SIMD instructions (e.g. [*i.e.*], multiply-accumulate operations) using one multiply-accumulate (MAC) unit while minimizing operational latency.

## BACKGROUND [OF THE INVENTION]

[0002] SIMD instructions are those instructions that perform the same operation on two or more pieces of a data word at the same time. A SIMD data word consists of two single-precision floating-point numbers, packed into a floating-point word. In an example of a 82-bit floating-point word, the low-SIMD data is stored in bits 31-0, and the high-SIMD data is stored in bits 63-32. [The r]Remaining bits (81-64) of the [example] 82-bit [bits (81-64)] word are set to a predefined constant.

[0003] Currently, two miscellaneous [31(A&B)] units 5, 6 and two MAC [41(A&B)] units 3, 4 are used [required] to perform SIMD instructions. [A m]Miscellaneous units [unit] (MISC) 5, 6 are devices [is a unit] that perform [performs] operations not requiring a multiply-accumulate functions, such as, [all] logical functions. A first MAC [41B] unit 3 is responsible for performing a multiply-accumulate operation on the high-bits of the SIMD word.[, and t] The second MAC [41A] unit 4 is responsible for performing a multiple-accumulate operation on the low-bits of the SIMD word. [The] MAC [41(A&B)] unit results are [then] forwarded to [combined on] a single register file 7 [21]. A block diagram of an example of the prior-art system architecture to perform SIMD instructions using multiple MAC [41(A&B)] units 3, 4 is illustrated in FIG. 1. This prior-art implementation, which includes [indicates] two full-precision [functional] MAC units 3, 4, further includes, two MISC [31(A&B)] units 5, 6, [MAC units (low 41A&high 41B) and two single-precision SIMD [MAC] units 1, 2 [11(A&B)]. The prior-art system architecture [It] can simultaneously perform any of two SIMD

instructions, one SIMD and one non-SIMD instruction, or two non-SIMD instructions.

[0004] Thus, a heretofore-unaddressed need exists in the industry to perform SIMD instructions using a single MAC unit while minimizing [the] operational latency.

## SUMMARY [OF THE INVENTION]

[0005] The present invention provides an apparatus and method for performing SIMD instructions (e.g. [i.e.], multiply-accumulate operations) using one MAC unit while minimizing [the] operational latency.

[0006] Briefly described, in architecture, an apparatus for performing single-instruction multiple-data instructions, includes a multiply-accumulate unit configured to generate a data result, the data result having a first half and a second half, a register communicatively coupled to the multiply-accumulate unit, the register configured to store the first half of the data result, and a miscellaneous-logic unit configured to initiate the release of the first half of the data result from the register to synchronize the first half of the data result with the second half of the data result [the apparatus can be implemented as follows. A MAC unit generates a first half of a data result in a first time period and a second half of a data result in a later time period. A deferred register stores the first half of a data result while the second half of the data result is being computed. A MISC logic is used to determine when to release the first half of a data result stored in the deferred register in order to synchronize the first half of a data result with the second half of a data result.]

[0007] The present invention can also be viewed as a method for performing SIMD instructions using one MAC unit while minimizing [the] operational latency. [In this regard, t]The method can be broadly summarized as follows [by the following steps]: providing a multiply-accumulate unit configured to generate a first half of a data result and a second half of a data result, applying the first half of the data result at an input of a register, and applying the first half of the data result and the second half of the data result at an input of a buffer when the first half of the data result and the second half of the data result are valid, otherwise applying an exception result at the input of the buffer the first half of the data result and the second half of the data result are invalid [A MAC unit generates a first operand data result. The first operand data result is input into a deferred register. The MAC unit generates a

second operand data result. While the MAC unit generates a first and second operand data result, an exception result is generated by a MISC logic. The MISC logic places the first operand data result and said second operand data result into a buffer if the MISC logic determines that the first operand data result and the second operand data result are valid. The MISC logic places the exception result into the buffer if the MISC logic determines that the first operand data result and the second operand data result are invalid].

[0008] Other features and advantages of the present invention will become apparent to one [with skill] skilled in the art upon examination of the following drawings and detailed description. It is intended that all such additional features and advantages be included herein within the scope of the present invention.

## BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The invention can be better understood with reference to the following drawings. The components in the drawings are not necessarily to scale, emphasis instead being placed upon clearly illustrating the principles of the present invention. Moreover, in the drawings, like reference numerals designate corresponding parts throughout the several views.

[0010] FIG. 1 is a block diagram of a prior-art system architecture capable of performing [to perform] SIMD instructions using multiple MAC units.

[0011] FIG. 2 is a block diagram of an embodiment of a system architecture capable of performing [to perform] SIMD instructions using a single MAC unit [of the present invention].

[0012] FIG. 3 is a flow chart of an embodiment [of a possible example for how data moves through the pipeline of] a method for processing data through the system architecture [to perform SIMD instructions using a single MAC unit as shown in] of FIG. 2.

[0013] FIG. 4 is an embodiment of a timing diagram [of a possible example for how data moves through the pipeline of a possible example of] illustrating how data is processed through the system architecture [to perform SIMD instructions using a single MAC unit as shown in] of FIG. 2.

**DETAILED DESCRIPTION [OF THE PREFERRED EMBODIMENT]**

[0014] Reference will now be made in detail to the description of the apparatus and method [invention] as illustrated in the drawings. While the apparatus and method [invention] will be described in connection with these drawings, there is no intent to limit it to the embodiment or embodiments disclosed herein [therein]. On the contrary, the intent is to cover all alternatives, modifications, and equivalents within the [spriit and] scope defined by the appended claims.

[0015] Illustrated in FIG. 2 is an embodiment [a possible example] of a [the] system architecture capable of performing [to perform] SIMD instructions using a [deferred] register and a single MAC unit [of the present invention]. As shown, [the] register file 21 provides operand [(A-C)] A, operand B, and operand C data on operand busses A 22, B 23, and C 24. Operand busses A 22, B 23 and C 24 transfer operand data from the register file 21 to logic 32 in MISC 31 and to the MAC 41. There is no logic in MISC 31 between register file 21 and MAC 41 for the operands. The MAC 41 receives [the] operand A, operand B, and operand C data on operand busses [A-C (22-24)] A 22, B 23, and C 24, respectively, in logic 42. Logic 42 also receives operational control codes [opcodes] from an [the] external control unit (not shown for simplicity of illustration). Operational control codes are used [that are necessary] to compute a desired [an appropriate] result.

[0016] [Using the operand (A-C) data and control opcodes,] MISC logic 32 uses the operand A, operand B, and operand C data and operational control codes to generate [generates] a series of four sets of result control signals and their complements to control the various bus drivers in both [the] MISC 31 and MAC 41. [The four sets of signals include:]

[0017] Result control signal A is generated in accordance with the following expression: A = miscop + macop * misc_result + simdop, where miscop indicates that there is an instruction for MISC 31, macop indicates that there is an instruction for MAC 41, misc_result indicates that there is a non-SIMD MAC 41 instruction that contains MISC 31 generated result(s), and simdop indicates that there is a SIMD instruction for either MISC 31 or MAC 41. Generating signal A configures [will enable the] data bus 36 to transmit data to [the] result data bus 71A. [The d]Data bus 36 [is able to transmit] transmits data to [the] result data bus 71A when signal 72 enables buffer/driver 33.

[0018] Result control signal B is generated in accordance with the following expression: B = miscop + macop * !simd * miscresult + macop * misc_result_high * simdhigh, where misc_result_high is a SIMD MAC 41 instruction with a MISC 31 result on the high-half data bits (i.e., bits 63-32), and simdhigh is the result of the SIMD operation on the high bits. Generating signal B configures [will enable the] data bus 37 to transmit data to the high-half result data bus 61A. [The d]Data bus 37 [is able to transmit] transmits data to the high-half result data bus 61A when signal 62 enables buffer/driver 34. The high-half result data bus 61A transmits data to [deferred] register 80 for storage. [The deferred r]Register 80 stores the first half of the [a] data result while the second half of the data result is being computed. [A] MISC logic 32 determines [is used to determine] when to release the first half of the [a] data result stored in [deferred] register 80 [in order] to synchronize the first half of the [a] data result with the second half of the data result.

[0019] Result control signal C is generated in accordance with the following expression: C = miscop + macop * !simd * miscresult + macop * misc_result_low * simdhigh, where misc_result_low is the MISC31 result on the low data bits (i.e., bits 31-0). Generating signal C configures [will enable the] data bus 38 to transmit data to the result data bus 51A. [The d]Data bus 38 [is able to transmit] transmits data to the result data bus 51A when signal 52 enables buffer/driver 35.

[0020] Result control signal D is generated in accordance with the following expression: D = macop * !misc_result_low * simdhigh. Generating signal D configures [will enable the] data bus 61B to transmit data from [deferred] register 80 to [the] result data bus 51B. [The d]Data bus 61B [is able to transmit] transmits data to [the] result data bus 51B when signal 75 enables buffer/driver 27. [These signals arrive during time x and time y on the timing diagram (FIG. 4). Basically, these signals can be considered FP4 signals that might be qualified by the simdhigh/low signals.]

[0021] Result control signals A-D are valid in MISC 31 and MAC 41 during period x and period y on the timing diagram (FIG. 4). Signals A-D are floating point control unit signals that are qualified by the simdhigh/simdlow signals. In SIMD mode, result control A is valid during period [time] y, result control B is valid for both periods [time] x and y, and result control signals D&C are valid for period

[time] y. In non-SIMD mode, result control signals A-D [these signals] are valid during floating-point clock stage [in] FP4.

[0022] [The four sets of signals listed above are comprised of] The result control signals listed above are generated in accordance with the following instructions and signals:

miscop = [there is] an instruction exists for [the] MISC 31 [unit];

macop = [there is] an instruction exists for [the] MAC 41 [unit];

misc_result = a non-SIMD MAC instruction that contains [has] results generated by MISC 31;

misc_result_low = a SIMD MAC instruction that contains [has] low-half data generated by [the] MISC 31 [unit];

misc_result_high = a SIMD MAC instruction that contains high-half data generated by [the] MISC 31 [unit];

simdhigh = asserted during clock stage FP4 (high-operand) during [cycle (the 2nd FP4) in] which, the results for the high-half SIMD are generated. (It is assumed that the signal simdhigh is only active when [if] signal simd is active);

simd = [there is] a SIMD instruction exists (for either MISC 31 or MAC 41).

[0023] These signals are generated by the MISC 31, based upon the operational control codes [opcodes] and operands. The operands are received by [the] MISC 31 from [the] register file 21. The operational control codes [opcodes] come from an external control unit (FPU Control) (not shown) that communicates with the main instruction fetch unit. The FPU Control and MISC 31 units are responsible for the correct staging of pipelined control information.

[0024] Bus drivers 27, 33, 34, and 35 [The bus drivers] in FIG. 2 will drive only when their enable [select] line is asserted. An example will be illustrated with regard to FIG. 3 herein described [defined] in detail below. The present apparatus is not limited to the bus control methodology illustrated and described in association with FIG. 2. Other methods of performing the bus multiplexing are possible, such as repeating multiplexers, etc. [Note that this diagram (FIG. 2) demonstrates only one functional unit. It is contemplated by the inventors that there can be two functional units, one on either side of the register file 21, similar to FIG. 1. The

configuration of the present invention (FIG. 2) is able to simultaneously perform two SIMD instructions (4 total operations), one SIMD and one non-SIMD instruction, or two non-SIMD instructions.]

[0025] Note that the apparatus illustrated in FIG. 2 contains one functional unit. Those skilled in the art should understand that a second functional unit can be arranged to interface with register file 21, as shown in the system architecture of FIG. 1. However, the apparatus illustrated in FIG. 2 is able to simultaneously perform two SIMD instructions (4 total operations), one SIMD and one non-SIMD instruction, or two non-SIMD instructions. While the embodiment illustrated in FIG. 2 requires an additional cycle of latency than that of the prior-art system illustrated in FIG. 1, the apparatus of FIG. 2 enables a SIMD instruction to be executed in parallel with another instruction (SIMD or non-SIMD). [While the implementation of the present invention (FIG. 2) requires an additional cycle of latency, that of FIG. 1, the system architecture of the present invention permits a SIMD instruction to be executed in parallel with another instruction (SIMD or non-SIMD). Of course, it is contemplated by the inventors that other methods of performing the bus multiplexing are possible, such as repeating multiplexers, etc.]

[0026] {Illustrated in] FIG. 3 is a flow chart of an embodiment [a possible example for how data moves through the pipeline of a possible example of] of a method for processing data through the system architecture [to perform SIMD instructions using a single MAC unit as shown in] of FIG. 2. First, the register file 21 (FIG. 2) drives data on operand busses A 22, B 23, and C 24, for two consecutive cycles. During the first cycle, as illustrated in block 101, MAC 41 latches the low-operand data into low-data latches in logic 42, which is now prepared to begin operations on the next cycle. Operational control code information arrives from the external control unit prior to or concurrently with the first clock cycle. [On the first cycle at step 101, the MAC 41 latches the low operand data into its high data latches in logic 42 and begins the operations on the next cycle. The opcode information also arrives from the external control unit at step 101.]

[0027] During [On] the second cycle, [at step 102] as illustrated in block 102, MAC 41 starts [will start operation] operations on the low-operand data and [latch] latches the high-operand data into the high-data latches of logic 42. [The] MISC 31 [will latch] latches both high and low-operand data [at step 102,] and operational control

codes [opcodes] arrive via busses [(22-24), again, from the external control unit] A 22, B 23, and C 24.

[0028] During [On] the third cycle, [at step 103] as illustrated in block 103, MAC 41 continues operation on the low-operand data and starts operation on the high-operand data. The MISC 31 begins its operation on both the high and low-operand data [at step 103]. A second instruction (either SIMD or non-SIMD) may have its operands and/or operational control codes [opcodes] delivered to the MISC 31, while [and] MAC 41 [units to start] starts on the next cycle.

[0029] During [On] the fourth cycle, [at step 104] as illustrated in block 104, MAC 41 continues operation on both the low [lower] and high [higher]-operand data. A third instruction can also enter the busses [(22-24)] A 22, B 23, and C 24 during [on] this cycle. This is a fully pipelined system and once the instructions leave a certain clock stage [state] (e.g., FP1, FP2, FP3, FP4, WRB) another SIMD or non-SIMD instruction can enter that clock stage.

[0030] During [On] the fifth cycle, [at step 105] as illustrated in block 105, MAC 41 delivers the low-operand data result onto the high-half result data bus 47. The low-operand data result is then transmitted to the high-half result data bus 61A. This is accomplished by applying [a using] signal 62 from logic 32 [in MSC 31] as an input at inverter 63 to generate enable signal 64. [This e]Enable signal 64 commands [authorizes] buffer/driver 44 to transmit [the] lower-operand data from the high-half result data bus 47 to the high-half result data bus 61A. Signal 62 is also input in its original value into buffer/driver 34. This original value for signal 62 disables buffer/driver 34 from transmitting [the] operand-data result from logic 32 [in MISC 31] onto high-half result data bus 61A. The low-operand data result from the high-half result data bus 61A is latched into [the deferred] register 80. Concurrently, during [Also in] the fifth cycle, MAC 41 continues to operate [operation] on the high-operand data.

[0031] During [On] the sixth cycle, as illustrated in block 106 [at step 106], [the] MISC 31 [will indicate] indicates whether to use the MAC 41 results or the MISC 31 exception [exceptional] results. [The] MISC 31 indicates which results are to be utilized by generating [the] signals on signal lines 52, 62, 72 and 75, respectively. These signals cause the appropriate bus drivers 25-27, 33-35, 43-45, 53, 63 and 73 to place result data on [the appropriate] result bus [(]51A, 61A, or 71A[)] as desired.

MISC 31 generates the following signals, illustrated in the table below, on signal lines 52, 62, 72, and 75, respectively, to command the appropriate bus drivers to place result data on result bus 51A, 61A, or 71A.

[0032] Cases 1-4 in Table I below are SIMD MAC operation cases. The cases are as follows:

| CASE | EXCEPTIONS | BUFFER/DRIVERS ON | SIGNALS ACTIVE |
|---|---|---|---|
| Case 1 | No exceptions | 27, 33, 44 [44,33] | 64, 72, 75 [75, 64, 72] |
| Case 2 | Low exception | 26, 33, 35, 44 [35, 26, 44, 33] | 52, 64, 72, not 75 [52 76 64, 72 |
| Case 3 | High exception | 27, 33, 44 [44,33] | 62, 72, 75 [75, 62, 72 |
| Case 4 | Both exceptions | 26, 33, 34, 35 [35, 26, 34, 33] | 52, 62, 72, not 75 [52, 76, 62, 72] |
| Case 5 | Non-SIMD, no exception | 26, 43, 44, 45 [45, 26, 44, 43] | 54, 64, 74, not 75 [54, 76, 64, 74] |
| Case 6 | Non-SIMD, exception | 26, 33, 34, 35 [35, 26, 34, 33] | 52, 72, not 75 [52, 76, 52, 72] |
| Case 7 | MISCOP | 26, 33, 34 [35, 26, 34, 33] | 62, 72, not 75 [52, 76, 62, 72] |

Table I

[0033] If the MISC 31 does not detect an exceptional case for the high mantissa, the MAC 41 delivers the high-operand data result onto the high-half result data bus 61A. If the MISC 31 does not detect an exceptional case for the low mantissa, [the deferred] register 80 drives the [be] lower-operand data result onto the lower-half result data bus 51A.

[0034] Whenever [the] MISC 31 detects an exception [exceptional case, it will be responsible for delivering], MISC 31 delivers the result. In any of the SIMD cases, [the] MISC 31 [will deliver] delivers the exponent result [from buffer/driver 33 by generating a signal on signal lines 72.] MISC 31 delivers the exponent result from buffer/driver 33 by generating a signal on signal line 72.

[0035] During [On] the seventh cycle, as illustrated in block 107 [at step 107], the combined result is [then] written to the register file 21 (FIG. 2). While the system architecture illustrated and described utilizes a four clock period latency other clock cycle latencies are possible. [While the example of the system architecture of the present invention utilizes a four cycle latency, it is contemplated by the inventors that other cycle latencies may e utilized.]

[0036] [Illustrated in] FIG. 4 is an embodiment of a timing diagram illustrating how data is processed through [of a possible example of how data moves through the pipeline of a possible example of] the system architecture [to perform SIMD instructions using a single MAC 41 as described above with regard to FIG. 3] of FIG. 2. As illustrated in FIG. 4 a plurality of non-SIMD operations 112 and SIMD operations 113 are controllably configured via result control signals A-D in accordance with low-operand data states 114 and high-operand data states 115 that correspond to clock signal trace 111. [First, the r]Register file 21 (FIG. 2) drives low-operand data [114] and high-operand data [115] on the operand busses A 22, B 23, and C24 (FIG. 2), over three [for two] consecutive clock cycles as shown by signal traces [signals] 131 and 132. The arrival of low-operand data, during clock stage FP1 for low-operand data is shown by signal trace 131. The arrival of high-operand data, during clock stage FP1 for high-operand data is indicated by signal trace 132. [On the first cycle, the MAC 41(FIG. 2) latches the low operand data 114 into its high data latches and begins the operations on the next cycle. The arrival of the low operand data is shown by signal 131.]

[0037] As indicated by signal trace 133, a low-operand result is calculated during clock stages FP2 and FP3 and latched during clock stage FP4 for low-operand data. Signal trace 134 illustrates that a high-operand result is calculated during clock stages FP2 and FP3 and latched during clock stage FP4 for high-operand data. The apparatus of FIG. 2 [On the fourth cycle, the MAC 41 continues operation on both the lower and higher operand data (114 at FP3 and 115 at FP2). This is] a fully pipelined system and once the instructions leave a clock state (e.g., FP1, FP2, FP3, FP4, WRB) another instruction, SIMD or non-SIMD can enter that clock stage. [The] MISC 31 generates signals 52, 62 and 72, which control the output buffer/drivers 43, 44 and 45 (FIG. 2). [These o]Output buffer/drivers 43, 44, and 45, control the data transmitted along [that gets put on] result data busses 51A, 61A, and 71A (FIG. 2), respectively during clock cycle 5 [in the fifth cycle].

[0038] During [On the fifth] clock cycle 5, [the] MAC 41 delivers the low-operand data result 133 onto the high-half result data bus 47 (FIG. 2). Signals on data busses 46, 47, and 48 (FIG. 2), [are] generated by MAC 41 are dependent upon the operand and the operational control codes [opcodes] applied on operand busses [22-24] A 22, B 23, and C 24. For SIMD instructions, high-half result data bus 47 is the most

significant output bus. For non-SIMD instructions, all three busses (i.e., 46, 47 and 48 (FIG. 2)) are significant output busses. [The l]Low-operand data result [133] is [then] transmitted to the high-half result data bus 61A. Thereafter, [The] low-operand data result [133] from the high-half result data bus 61A is latched into [the deferred] register 80 (FIG. 2). Also, during [in the fifth] clock cycle 5, [the] MAC 41 (FIG. 2) continues to operate [operation] on the high-operand data [(115 at FP3)] as indicated by signal trace 134.

[0039] During [On the sixth] clock cycle 6, [the] MAC 41 (FIG. 2) delivers the high-operand data result as indicated in signal trace 134 onto the high-half result data bus 47. [The] MISC 31 indicates [will indicate] whether [to use the] MAC 41 results or the MISC 31 exception [exceptional] results are applied to high-half result data bus 47. In addition, SIMD results are now available as illustrated by signal trace 136.

[0040] During [On the seventh] clock cycle 7, the combined result is [then] written to the register file 21 (FIG. 2) as indicated by signal trace 135.

[0041] It should be emphasized that the above-described embodiments of the present invention, particularly, any "preferred" embodiments, are merely possible examples of implementations, merely set forth for a clear understanding of the principles of the invention. Many variations and modifications may be made to the above-described embodiment(s) of the invention without departing substantially from the principles of the invention. All such modifications and variations are intended to be included herein within the scope of the present invention and protected by the following claims.

## APPENDIX A - AMENDMENTS TO THE CLAIMS

The following claims have been amended by deleting the bracketed ("[ ]") portions and adding the underlined ("__") portions.

1. (Amended) An apparatus for performing single-instruction multiple-data [Single Instruction Multiple Data (SIMD)] instructions using a single multiply-accumulate (MAC) unit, [while minimizing the operational latency, said apparatus] comprising:

a MAC unit[, said MAC unit generates] configured to generate a data result, the data result having a first half [of a data result] and a second half [of said data result];

a [defer] register communicatively coupled to the multiply-accumulate unit, the register configured to store [stores] the [said] first half of the [a] data result; and

a miscellaneous-logic unit[, said MISC unit determines when] configured to initiate the release of the [said] first half of the [a] data result from the [stored in said defer] register to synchronize the [said] first half of the [a] data result with the [said] second half of the [said] data result.

2. (Amended) The apparatus of claim 1, wherein said MAC unit generates the [said] first half of the [a] data result before the [said] second half of the [said] data result.

3. (Amended) The apparatus of claim 2, further comprising:

a register file[, wherein said register file receives] configured to receive the [said] first half of the [said] data result substantially concurrently [currently] with the [said] second half of the [said] data result.

4. (Amended) The apparatus of claim 3, wherein said miscellaneous-logic [MISC] unit generates an exception result when [if] said miscellaneous-logic [MISC] unit determines the [said] first half of the [said] data result is erroneous [in error].

5. (Amended) The apparatus of claim 4, wherein said miscellaneous-logic [MISC] unit is configured to initiate the release of one of the [further determines if said] first half of the [a] data result stored in said [defer] register or initiate the release of the [said] exception result [is to be release to said register file].

6. (Amended) A method for performing single-instruction multiple-data [Single Instruction Multiple Data (SIMD)] instructions using a single multiply-accumulate unit [while minimizing the operational latency], comprising [the steps of]:

providing a multiply-accumulate unit configured to generate [generating] a first half of a [operand] data result [by said MAC unit] and a second half of a data result;

applying the first half of the data result at an input of [inputting said first operand data result to] a [deferred] register;

[generating a second operand data result by said MAC unit;]

using a miscellaneous-logic unit to generate [generating] an exception result [by a MISC unit]; and

applying the first half of the data result and the second half of the data result at an input of [inputting said first operand data result and said second operand data result into] a buffer when [if] the [said MISC] miscellaneous-logic unit determines that the [said] first half of the [operand] data result and the [said] second half of the [operand] data result are valid, otherwise [; and] applying an exception result at the input of the [inputting said exception result into said] buffer [if] when the [said MISC] miscellaneous-logic unit determines that said] the first half of the [operand] data result and the [said] second half of the [operand] data result are invalid.

7. (Amended) The method of claim 6, further comprising [the steps of]:

latching a first operand [data] into the [said MAC] multiply-accumulate unit; and

latching a second operand [data] into the [said MAC] multiply-accumulate unit.

8. (Amended) The method of claim 7, further comprising [the steps of]:

generating the [said] first half of the [operand] data result from the [said] first operand; and

generating the [said] second half of the [operand] data result from the [a] second operand [data].

9. (Amended) The method of claim 8 [6], further comprising [the step of:]:

latching the [a] first half of the data result in a miscellaneous logic [operand and a second operand data into said MISC] unit; and

latching the second half of the data result in the miscellaneous logic unit.

10. (Amended) The method of claim 9, further comprising [the step of:]:

generating the [said] exception result from the [said] first half of the data result [operand] and the [said] second half of the data result [operand data].

11. (Amended) An apparatus for performing single-instruction multiple-data [Single Instruction Multiple Data (SIMD)] instructions, [using a single multiply-accumulate (MAC) unit while minimizing the operational latency, said apparatus] comprising:

means for generating a first [operand] data result responsive to a first operand;

means for storing [inputting] the [said] first [operand] data result [to a deferred register];

means for generating a second [operand] data result responsive to a second operand;

means for generating an exception result responsive to the first and second data results;

means for forwarding the [inputting said] first [operand] data result [from said deferred register] and the [said] second [operand] data result [into] to a buffer [if said] when the exception result [generating means determines] indicates that the [said] first [operand] data result and the [said] second [operand] data result are valid; and

means for communicating [inputting] the [said] exception to the buffer when the means for forwarding indicates that the first and second data results [result into said buffer if said exception result generating means determines that said first operand data result and said second operand data result] are invalid.

12. (Amended) The apparatus of claim 11, further comprising:

means for storing [latching] the [a] first operand [data into said first operand data result generating means]; and

means for storing [latching] the [a] second operand [data into said second operand data result generating means].

## APPENDIX D - AMENDMENTS TO THE DRAWINGS

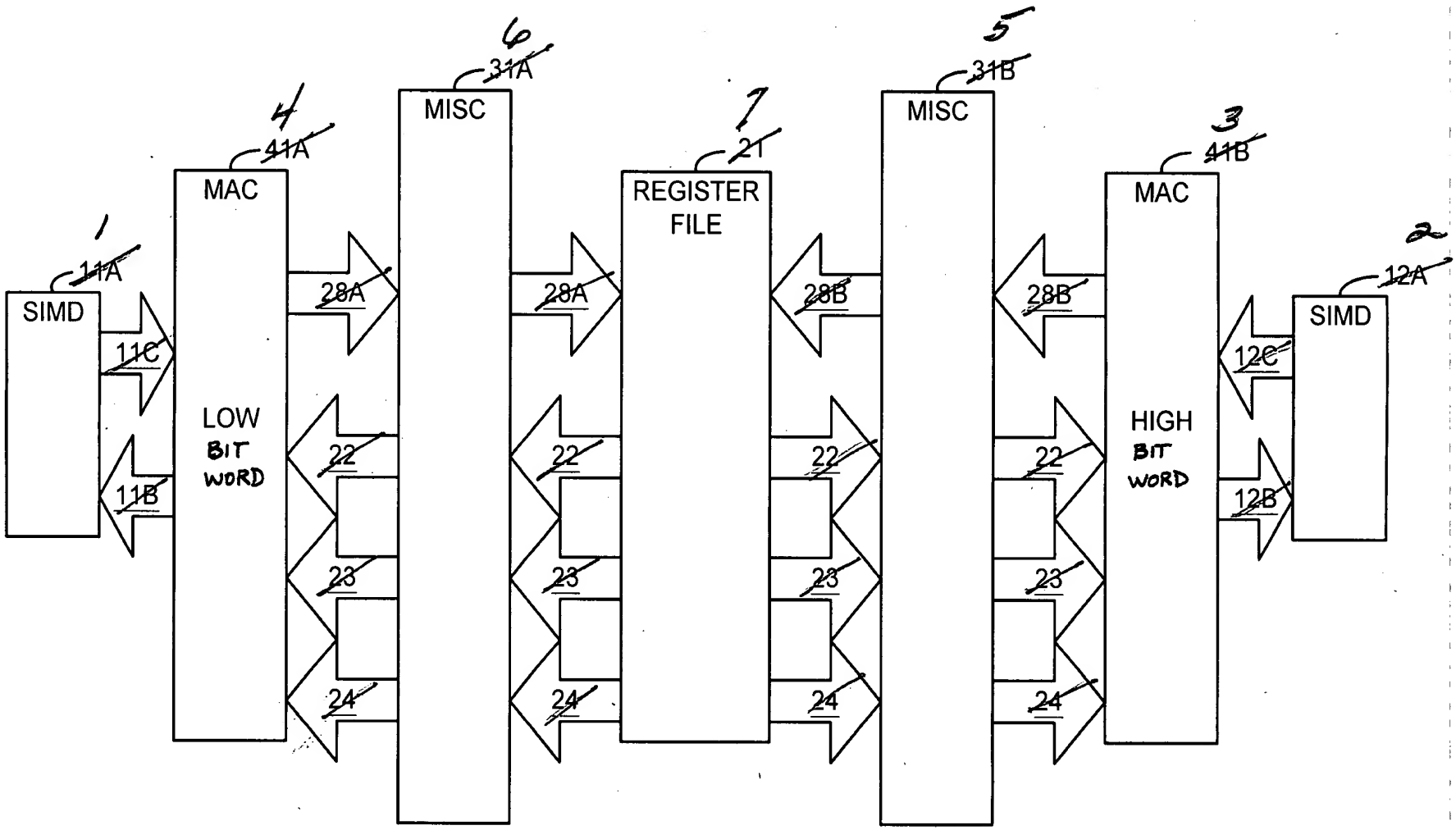
Please find enclosed a redlined copy showing the proposed changes to the drawings.

DRAWING CHANGES
APPROVED TM 2/25/03

PATENT & TRADEMARK OFFICE
DEC 30 2002
JC54

# FIG. 1

PRIOR ART



DEC 30 2002
PATENT & TRADEMARK OFFICE

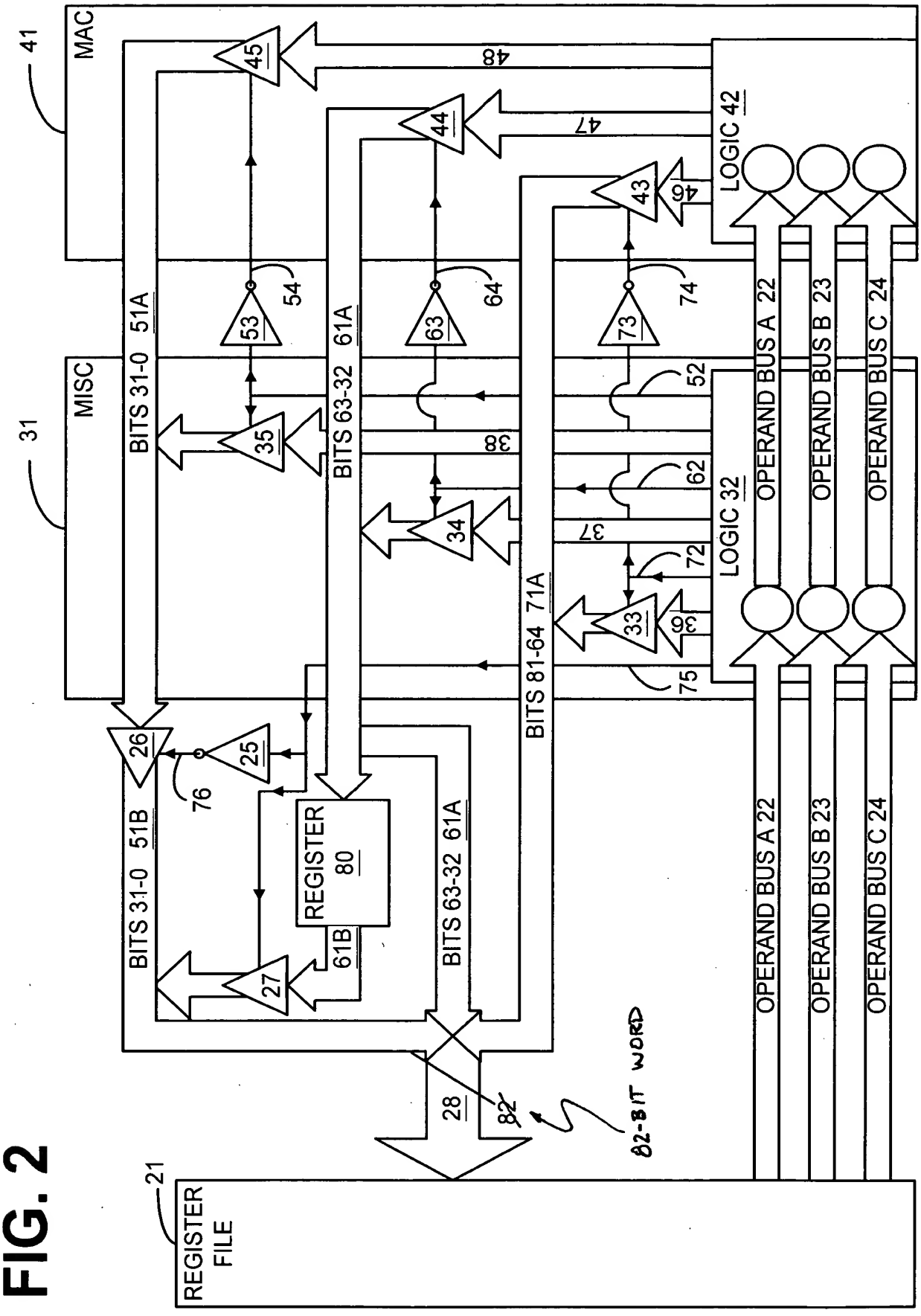
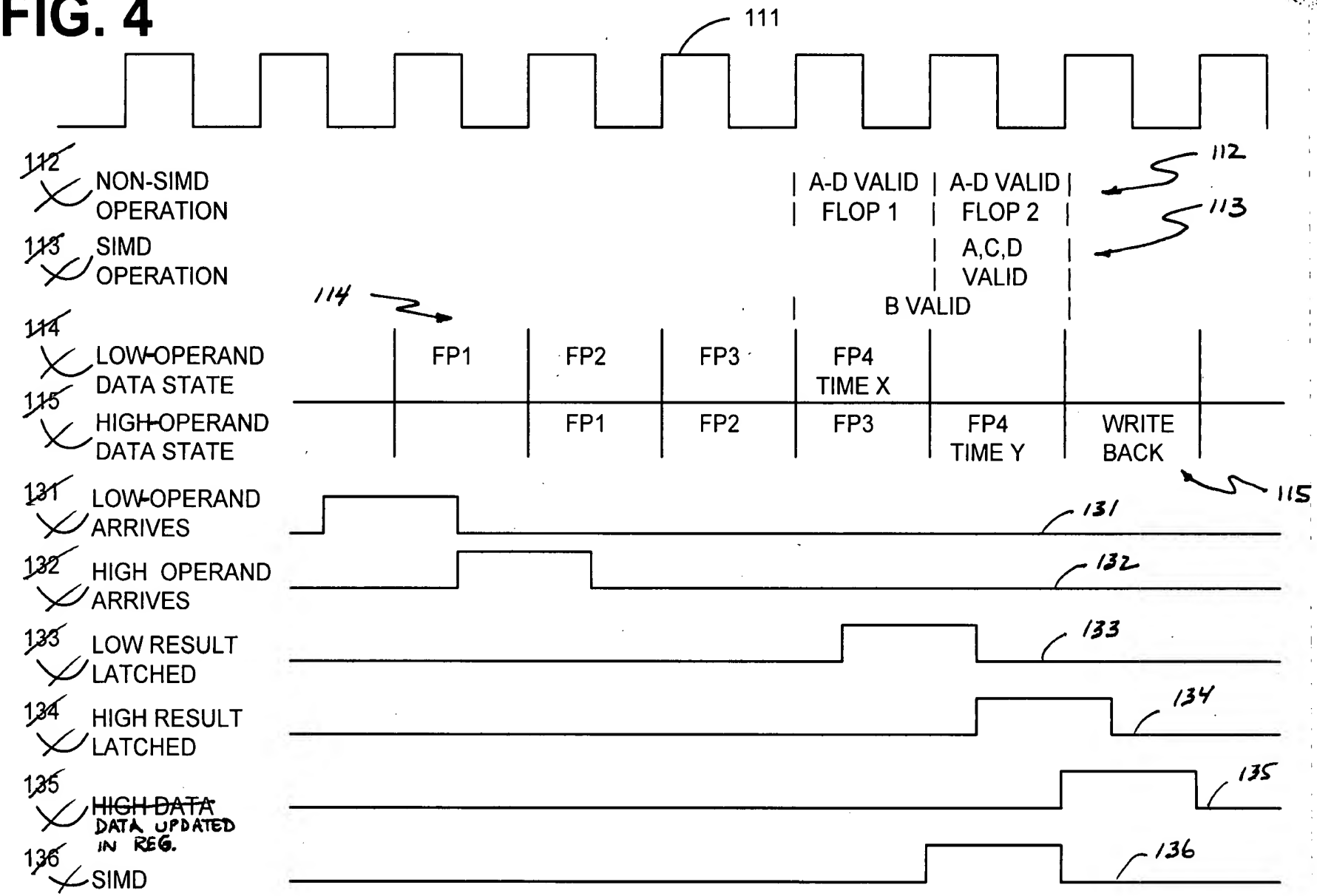



FIG. 2

PATENT & TRADEMARK OFFICE
DEC 30 2002

# FIG. 4